Amendments to the Specification:

The paragraph beginning on line 20 of page 4 has been amended as shown:

The present invention utilizes an encoding technique for instruction codes in a VLIW that reduces the instruction memory requirements through the use of an enable signal and action signal for each instruction. In a preferred embodiment, the aspects of the present invention are provided in the context of an adaptable computing engine in accordance with the description in co-pending U.S. Patent application, serial no. ~~_____~~ 09/815,122, entitled "Adaptive Integrated Circuitry with Heterogeneous and Reconfigurable Matrices of Diverse and Adaptive Computational Units Having Fixed, Application Specific Computational Elements," assigned to the assignee of the present invention and incorporated by reference in its entirety herein. Portions of that description are reproduced hereinbelow for clarity of presentation of the aspects of the present invention. It should be appreciated that although the aspects are described with particular reference and with particular applicability to the adaptable computing engine environment, this is meant as illustrative and not restrictive of a preferred embodiment.

The paragraph beginning on line 6 of page 10 has been amended as shown:

Referring, then, to Figure 3a, as an initial step in the processing of an algorithm into instruction code, the algorithm is defined mathematically. In the example shown, a value, x[i], is summed over the range i=0 to j, where j ranges from 0 to N-1, and N=7, to produce an output value y[j]. Once defined, the algorithm is written as a program in a programming language appropriate for the computation unit, which for the ACE is preferably the Q programming language. ~~The Q programming language is presented in more detail in co-pending U.S. Patent application, serial no.[Docket No. QST-009-US], filed _____, entitled *Q Programming Language*, and assigned to the assignee of the present invention.~~ Figure 3b illustrates a Q program for the example algorithm shown in Figure 3a.

The paragraph beginning on line 14 of page 11 has been amended as shown:

Figure 3c illustrates the dataflow graph for the example program shown in Figure 3b. In order to perform the operations represented by the dataflow graph, the graph is scheduled in time and assigned to hardware resources in space by a scheduler. Co-pending U.S. Patent application, serial no. ~~(Docket No. 2096P)~~ 09/872,397, filed May 31, 2001, entitled *Method and System for Scheduling in an Adaptive Computing Engine* and assigned to the assignee of the present invention, presents a preferred embodiment of a scheduler and its description is incorporated herein by reference. In general, the scheduler determines which nodes in the list of nodes specified by the input dataflow graph can be executed in parallel on a single clock cycle and which nodes must be delayed to subsequent cycles. The scheduler further assigns registers to hold intermediate values (as required by the delayed execution of nodes), to hold state variables, and to hold constants. In addition, the scheduler analyzes register life to determine when registers can be reused, allocates nodes to computation units, and schedules nodes to execute on specific clock cycles. Thus, for each node, there are several specifications, including: an operational code (Op Code), a pointer to the source code (e.g., firFilter.q, line 55); a pre-assigned computation unit, if any; a list of input edges; a list of output edges; and for each edge, a source node, a destination node, and a state flag, i.e., a flag that indicates whether the edge has an initial value.